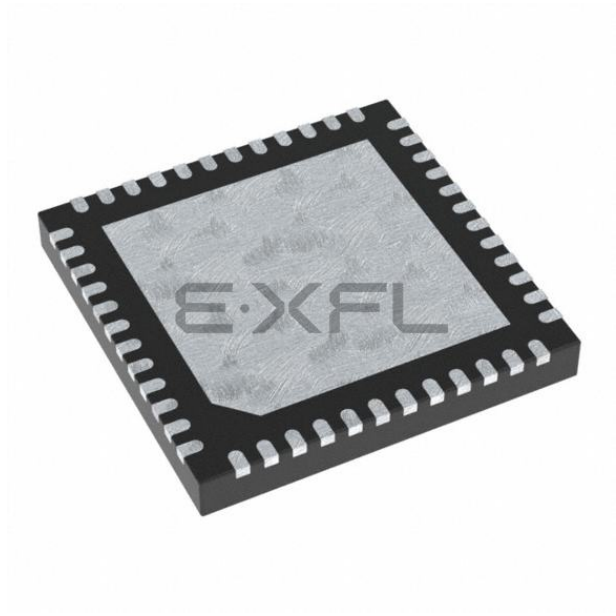


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	ARM® Cortex®-M0+
Core Size	32-Bit Single-Core
Speed	48MHz
Connectivity	I²C, LINbus, SCI, SPI, UART/USART, USB
Peripherals	DMA, POR, PWM, WDT
Number of I/O	37
Program Memory Size	256KB (256K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	32K x 8
Voltage - Supply (Vcc/Vdd)	1.62V ~ 3.63V
Data Converters	A/D 14x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	48-VFQFN Exposed Pad
Supplier Device Package	48-QFN (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsaml21g18b-mnt

Bit 2 – RSTC: Interrupt Flag for RSTC

Bit 1 – MCLK: Interrupt Flag for MCLK

Bit 0 – PM: Interrupt Flag for PM

15. DSU - Device Service Unit

15.1. Overview

The Device Service Unit (DSU) provides a means to detect debugger probes. This enables the ARM Debug Access Port (DAP) to have control over multiplexed debug pads and CPU reset. The DSU also provides system-level services to debug adapters in an ARM debug system. It implements a CoreSight Debug ROM that provides device identification as well as identification of other debug components within the system. Hence, it complies with the ARM Peripheral Identification specification. The DSU also provides system services to applications that need memory testing, as required for IEC60730 Class B compliance, for example. The DSU can be accessed simultaneously by a debugger and the CPU, as it is connected on the High-Speed Bus Matrix. For security reasons, some of the DSU features will be limited or unavailable when the device is protected by the NVMCTRL security bit.

Related Links

NVMCTRL – Non-Volatile Memory Controller on page 489

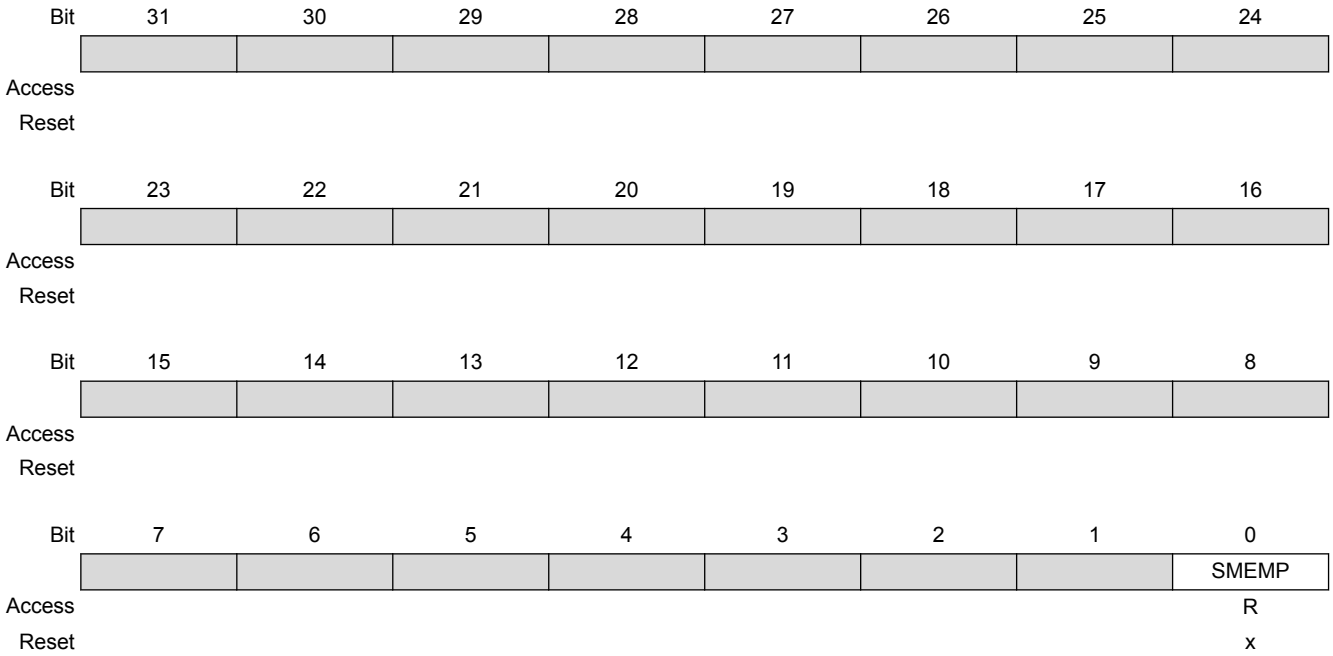
Security Bit on page 497

15.2. Features

- CPU reset extension
- Debugger probe detection (Cold- and Hot-Plugging)
- Chip-Erase command and status
- 32-bit cyclic redundancy check (CRC32) of any memory accessible through the bus matrix
- ARM® CoreSight™ compliant device identification
- Two debug communications channels
- Debug access port security filter
- Onboard memory built-in self-test (MBIST)

15.13.13. CoreSight ROM Table Memory Type

Name: MEMTYPE
Offset: 0x1FCC
Reset: 0x0000000X
Property: -



Bit 0 – SMEMP: System Memory Present
This bit indicates whether system memory is present on the bus that connects to the ROM table.

This bit is set at power-up if the device is not protected, indicating that the system memory is accessible from a debug adapter.

This bit is cleared at power-up if the device is protected, indicating that the system memory is not accessible from a debug adapter.

Offset	Name	Bit Pos.								
0x0100	PCHCTRL32	7:0	WRTLOCK	CHEN			GEN[3:0]			
0x0101		15:8								
0x0102		23:16								
0x0103		31:24								
0x0104	PCHCTRL33	7:0	WRTLOCK	CHEN			GEN[3:0]			
0x0105		15:8								
0x0106		23:16								
0x0107		31:24								
0x0108	PCHCTRL34	7:0	WRTLOCK	CHEN			GEN[3:0]			
0x0109		15:8								
0x010A		23:16								
0x010B		31:24								

17.8. Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Optional PAC write-protection is denoted by the "PAC Write-Protection" property in each individual register description. For details, refer to Register Access Protection.

Some registers are synchronized when read and/or written. Synchronization is denoted by the "Write-Synchronized" or the "Read-Synchronized" property in each individual register description. For details, refer to Synchronization.

18.8.5. CPU Clock Division

Name: CPUDIV
Offset: 0x05
Reset: 0x01
Property: PAC Write-Protection

Bit	7	6	5	4	3	2	1	0
	CPUDIV[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

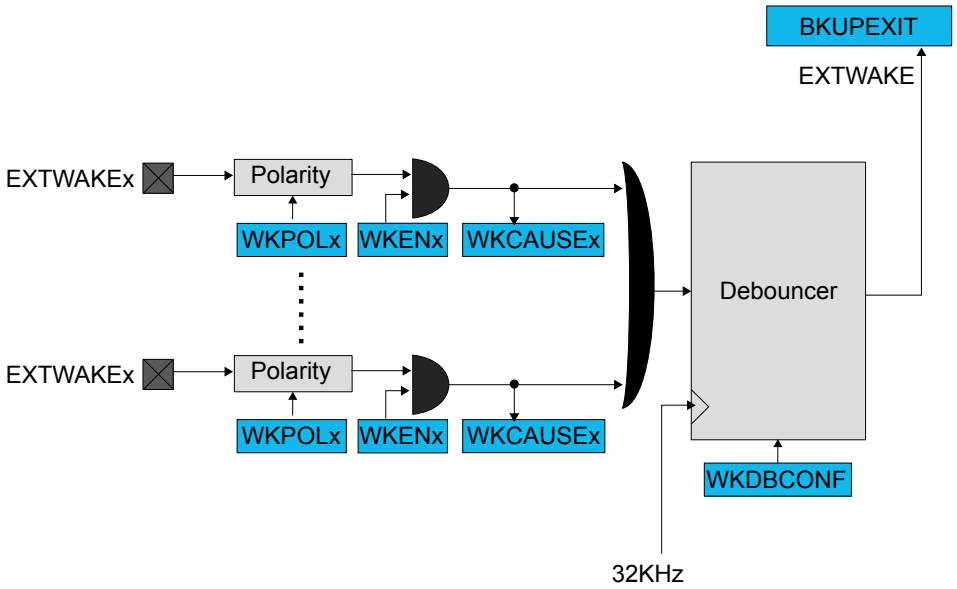
Bits 7:0 – CPUDIV[7:0]: CPU Clock Division Factor
These bits define the division ratio of the main clock prescaler related to the CPU clock domain.

To ensure correct operation, frequencies must be selected so that $F_{CPU} \geq F_{LP}$ (i.e. LPDIV ≥CPUDIV).

Frequencies must never exceed the specified maximum frequency for each clock domain.

Value	Name	Description
0x01	DIV1	Divide by 1
0x02	DIV2	Divide by 2
0x04	DIV4	Divide by 4
0x08	DIV8	Divide by 8
0x10	DIV16	Divide by 16
0x20	DIV32	Divide by 32
0x40	DIV64	Divide by 64
0x80	DIV128	Divide by 128
others	-	Reserved

Figure 19-2. External Wake-up Block Diagram



Related Links

OSC32KCTRL – 32KHz Oscillators Controller on page 273

19.6.2.4. Reset Causes and Effects

The latest Reset cause is available in RCAUSE register, and can be read during the application boot sequence in order to determine proper action.

These are the groups of Reset sources:

- Power supply Reset: Resets caused by an electrical issue. It covers POR and BODs Resets
- User Reset: Resets caused by the application. It covers external Resets, system Reset requests and watchdog Resets
- Backup reset: Resets caused by a Backup Mode exit condition

The following table lists the parts of the device that are reset, depending on the Reset type.

Table 19-1. Effects of the Different Reset Causes

	Power Supply Reset		User Reset		Backup Reset
	POR, BOD33	BOD12	External Reset	WDT Reset, System Reset Request	RTC, EXTWAKE, BBPS
RTC, OSC32KCTRL, RSTC, CTRLA.IORET bit of PM	Y	N	N	N	N
GCLK with WRTLOCK	Y	Y	N	N	Y
Debug logic	Y	Y	Y	N	Y
Others	Y	Y	Y	Y	Y

The external Reset is generated when pulling the $\overline{\text{RESET}}$ pin low.

Increment Step Size bit group in the Block Transfer Control register (BTCTRL.STEPSIZE). If BTCTRL.STEPSEL=0, the step size for the source incrementation will be the size of one beat.

When source address incrementation is configured (BTCTRL.SRCINC=1), SRCADDR is calculated as follows:

If BTCTRL.STEPSEL=1:

$$\text{SRCADDR} = \text{SRCADDR}_{START} + BTCNT \cdot \left(BEATSIZE + 1\right) \cdot 2^{\text{STEPSIZE}}$$

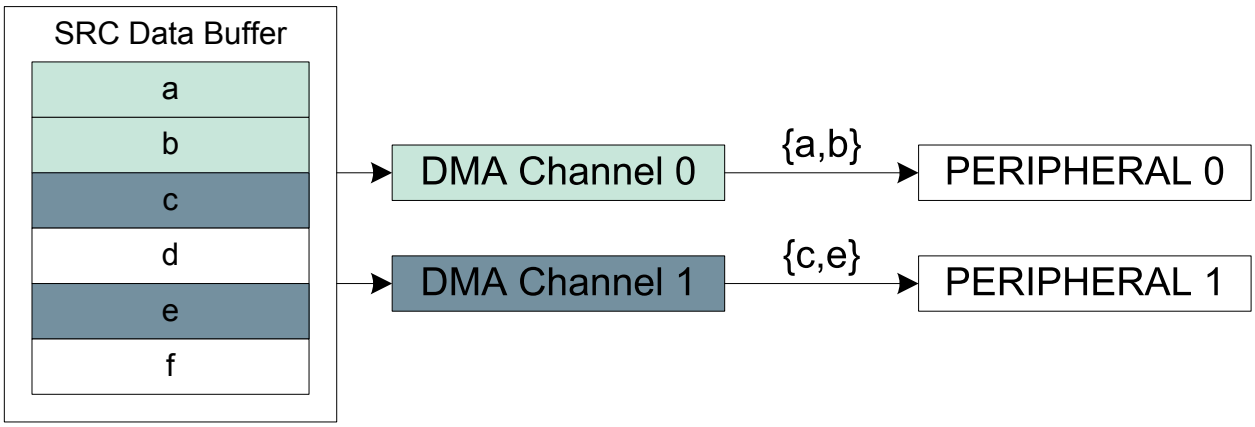
If BTCTRL.STEPSEL=0:

$$\text{SRCADDR} = \text{SRCADDR}_{START} + BTCNT \cdot (BEATSIZE + 1)$$

- $\text{SRCADDR}_{\text{START}}$ is the source address of the first beat transfer in the block transfer
- BTCNT is the initial number of beats remaining in the block transfer
- BEATSIZE is the configured number of bytes in a beat
- STEPSIZE is the configured number of beats for each incrementation

The following figure shows an example where DMA channel 0 is configured to increment the source address by one beat after each beat transfer (BTCTRL.SRCINC=1), and DMA channel 1 is configured to increment the source address by two beats (BTCTRL.SRCINC=1, BTCTRL.STEPSEL=1, and BTCTRL.STEPSIZE=0x1). As the destination address for both channels are peripherals, destination incrementation is disabled (BTCTRL.DSTINC=0).

Figure 26-8. Source Address Increment



Incrementation for the destination address of a block transfer is enabled by setting the Destination Address Incrementation Enable bit in the Block Transfer Control register (BTCTRL.DSTINC=1). The step size of the incrementation is configurable by clearing BTCTRL.STEPSEL=0 and writing BTCTRL.STEPSIZE to the desired step size. If BTCTRL.STEPSEL=1, the step size for the destination incrementation will be the size of one beat.

When the destination address incrementation is configured (BTCTRL.DSTINC=1), SRCADDR must be set and calculated as follows:

$DSTADDR = DSTADDR_{START} + BTCNT \bullet \left(BEATSIZE + 1\right) \bullet 2^{STEPSIZE}$	where BTCTRL.STEPSEL is zero
$DSTADDR = DSTADDR_{START} + BTCNT \bullet (BEATSIZE + 1)$	where BTCTRL.STEPSEL is one

- $\text{DSTADDR}_{\text{START}}$ is the destination address of the first beat transfer in the block transfer
- BTCNT is the initial number of beats remaining in the block transfer

26.9. Register Summary - LP SRAM

Offset	Name	Bit Pos.								
0x00	BTCTRL	7:0				BLOCKACT[1:0]		EVOSEL[1:0]		VALID
0x01		15:8	STEPSIZE[2:0]			STEPSEL	DSTINC	SRCINC	BEATSIZE[1:0]	
0x02	BTCNT	7:0	BTCNT[7:0]							
0x03		15:8	BTCNT[15:8]							
0x04	SRCADDR	7:0	SRCADDR[7:0]							
0x05		15:8	SRCADDR[15:8]							
0x06		23:16	SRCADDR[23:16]							
0x07		31:24	SRCADDR[31:24]							
0x08	DSTADDR	7:0	DSTADDR[7:0]							
0x09		15:8	DSTADDR[15:8]							
0x0A		23:16	DSTADDR[23:16]							
0x0B		31:24	DSTADDR[31:24]							
0x0C	DESCADDR	7:0	DESCADDR[7:0]							
0x0D		15:8	DESCADDR[15:8]							
0x0E		23:16	DESCADDR[23:16]							
0x0F		31:24	DESCADDR[31:24]							

26.10. Register Description - LP SRAM

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Optional PAC write-protection is denoted by the "PAC Write-Protection" property in each individual register description. For details, refer to Register Access Protection.

Some registers are enable-protected, meaning they can only be written when the peripheral is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

Offset	Name	Bit Pos.								
0x2C	EVCTRL	7:0	PORTEI0	EVACT0[1:0]		PID0[4:0]				
0x2D		15:8	PORTEI1	EVACT1[1:0]		PID1[4:0]				
0x2E		23:16	PORTEI2	EVACT2[1:0]		PID2[4:0]				
0x2F		31:24	PORTEI3	EVACT3[1:0]		PID3[4:0]				
0x30	PMUX0	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x31	PMUX1	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x32	PMUX2	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x33	PMUX3	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x34	PMUX4	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x35	PMUX5	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x36	PMUX6	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x37	PMUX7	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x38	PMUX8	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x39	PMUX9	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x3A	PMUX10	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x3B	PMUX11	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x3C	PMUX12	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x3D	PMUX13	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x3E	PMUX14	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x3F	PMUX15	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x40	PINCFG0	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x41	PINCFG1	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x42	PINCFG2	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x43	PINCFG3	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x44	PINCFG4	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x45	PINCFG5	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x46	PINCFG6	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x47	PINCFG7	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x48	PINCFG8	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x49	PINCFG9	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x4A	PINCFG10	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x4B	PINCFG11	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x4C	PINCFG12	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x4D	PINCFG13	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x4E	PINCFG14	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x4F	PINCFG15	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x50	PINCFG16	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x51	PINCFG17	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x52	PINCFG18	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x53	PINCFG19	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x54	PINCFG20	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x55	PINCFG21	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x56	PINCFG22	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x57	PINCFG23	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x58	PINCFG24	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x59	PINCFG25	7:0		DRVSTR				PULLEN	INEN	PMUXEN
0x5A	PINCFG26	7:0		DRVSTR				PULLEN	INEN	PMUXEN

The baud-rate generator is capable of running on the GCLK_SERCOMx_CORE clock or an external clock.

Address matching logic is included for SPI and I^2C operation.

31.6.2. Basic Operation

31.6.2.1. Initialization

The SERCOM must be configured to the desired mode by writing the Operating Mode bits in the Control A register (CTRLA.MODE). Refer to table SERCOM Modes for details.

Table 31-1. SERCOM Modes

CTRLA.MODE	Description
0x0	USART with external clock
0x1	USART with internal clock
0x2	SPI in slave operation
0x3	SPI in master operation
0x4	I^2C slave operation
0x5	I^2C master operation
0x6-0x7	Reserved

For further initialization information, see the respective SERCOM mode chapters:

Related Links

SERCOM USART – SERCOM Universal Synchronous and Asynchronous Receiver and Transmitter on page 577
SERCOM SPI – SERCOM Serial Peripheral Interface on page 615
SERCOM I2C – SERCOM Inter-Integrated Circuit on page 648

31.6.2.2. Enabling, Disabling, and Resetting

This peripheral is enabled by writing '1' to the Enable bit in the Control A register (CTRLA.ENABLE), and disabled by writing '0' to it.

Writing ‘1’ to the Software Reset bit in the Control A register (CTRLA.SWRST) will reset all registers of this peripheral to their initial states, except the DBGCTRL register, and the peripheral is disabled.

Refer to the CTRLA register description for details.

31.6.2.3. Clock Generation – Baud-Rate Generator

The baud-rate generator, as shown in Figure 31-3, generates internal clocks for asynchronous and synchronous communication. The output frequency (f_{BAUD}) is determined by the Baud register (BAUD) setting and the baud reference frequency (f_{ref}). The baud reference clock is the serial engine clock, and it can be internal or external.

For asynchronous communication, the /16 (divide-by-16) output is used when transmitting, whereas the /1 (divide-by-1) output is used while receiving.

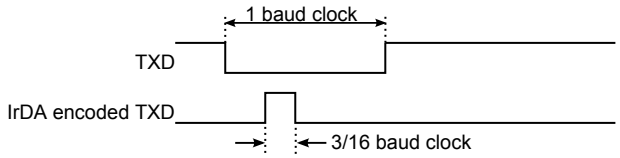
For synchronous communication, the /2 (divide-by-2) output is used.

This functionality is automatically configured, depending on the selected operating mode.

- and 16x sample rate (CTRLA.SAMPR[0]=0).

During transmission, each low bit is transmitted as a high pulse. The pulse width is 3/16 of the baud rate period, as illustrated in the figure below.

Figure 32-10. IrDA Transmit Encoding



The reception decoder has two main functions.

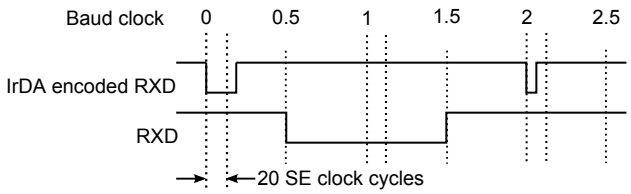
The first is to synchronize the incoming data to the IrDA baud rate counter. Synchronization is performed at the start of each zero pulse.

The second main function is to decode incoming Rx data. If a pulse width meets the minimum length set by configuration (RXPL.RXPL), it is accepted. When the baud rate counter reaches its middle value (1/2 bit length), it is transferred to the receiver.

Note: Note that the polarity of the transmitter and receiver are opposite: During transmission, a '0' bit is transmitted as a '1' pulse. During reception, an accepted '0' pulse is received as a '0' bit.

Example: The figure below illustrates reception where RXPL.RXPL is set to 19. This indicates that the pulse width should be at least 20 SE clock cycles. When using BAUD=0xE666 or 160 SE cycles per bit, this corresponds to 2/16 baud clock as minimum pulse width required. In this case the first bit is accepted as a '0', the second bit is a '1', and the third bit is also a '1'. A low pulse is rejected since it does not meet the minimum requirement of 2/16 baud clock.

Figure 32-11. IrDA Receive Decoding



32.6.3.4. Break Character Detection and Auto-Baud

Break character detection and auto-baud are available in this configuration:

- Auto-baud frame format (CTRLA.FORM = 0x04 or 0x05),
- Asynchronous mode (CTRLA.CMODE = 0),
- and 16x sample rate using fractional baud rate generation (CTRLA.SAMPR = 1).

The auto-baud follows the LIN format. All LIN Frames start with a Break Field followed by a Sync Field. The USART uses a break detection threshold of greater than 11 nominal bit times at the configured baud rate. At any time, if more than 11 consecutive dominant bits are detected on the bus, the USART detects a Break Field. When a Break Field has been detected, the Receive Break interrupt flag (INTFLAG.RXBRK) is set and the USART expects the Sync Field character to be 0x55. This field is used to update the actual baud rate in order to stay synchronized. If the received Sync character is not 0x55, then the Inconsistent Sync Field error flag (STATUS.ISF) is set along with the Error interrupt flag (INTFLAG.ERROR), and the baud rate is unchanged.

when a stop condition is detected, or when a time-out occurs (inactive bus time-out needs to be configured).

If a start condition is generated internally by writing the Address bit group in the Address register (ADDR.ADDR) while IDLE, the OWNER state (0b10) is entered. If the complete transaction was performed without interference, i.e., arbitration was not lost, the I^{2}C master can issue a stop condition, which will change the bus state back to IDLE.

However, if a packet collision is detected while in OWNER state, the arbitration is assumed lost and the bus state becomes BUSY until a stop condition is detected. A repeated start condition will change the bus state only if arbitration is lost while issuing a repeated start.

Regardless of winning or losing arbitration, the entire address will be sent. If arbitration is lost, only 'ones' are transmitted from the point of losing arbitration and the rest of the address length.

Note: Violating the protocol may cause the I^{2}C to hang. If this happens it is possible to recover from this state by a software reset (CTRLA.SWRST='1').

Related Links

CTRLA on page 692

34.6.2.4. I^{2}C Master Operation

The I^{2}C master is byte-oriented and interrupt based. The number of interrupts generated is kept at a minimum by automatic handling of most events. The software driver complexity and code size are reduced by auto-triggering of operations, and a special smart mode, which can be enabled by the Smart Mode Enable bit in the Control A register (CTRLA.SMEN).

The I^{2}C master has two interrupt strategies.

When SCL Stretch Mode (CTRLA.SCLSM) is '0', SCL is stretched before or after the acknowledge bit . In this mode the I^{2}C master operates according to Master Behavioral Diagram (SCLSM=0). The circles labelled "M*n*" (M1, M2..) indicate the nodes the bus logic can jump to, based on software or hardware interaction.

This diagram is used as reference for the description of the I^{2}C master operation throughout the document.

34.10.9. Address

Name: ADDR
Offset: 0x24
Reset: 0x0000
Property: Write-Synchronized

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	LEN[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TENBITEN	HS	LENEN			ADDR[2:0]		
Access	R/W	R/W	R/W			R/W	R/W	R/W
Reset	0	0	0			0	0	0

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bits 23:16 – LEN[7:0]: Transaction Length
These bits define the transaction length of a DMA transaction from 0 to 255 bytes. The Transfer Length Enable (LENEN) bit must be written to '1' in order to use DMA.

Bit 15 – TENBITEN: Ten Bit Addressing Enable
This bit enables 10-bit addressing. This bit can be written simultaneously with ADDR to indicate a 10-bit or 7-bit address transmission.

Value	Description
0	10-bit addressing disabled.
1	10-bit addressing enabled.

Bit 14 – HS: High Speed
This bit enables High-speed mode for the current transfer from repeated START to STOP. This bit can be written simultaneously with ADDR for a high speed transfer.

Value	Description
0	High-speed transfer disabled.
1	High-speed transfer enabled.

Bit 13 – LENEN: Transfer Length Enable

Writing a '1' to an Event Output bit in the Event Control register (EVCTRL.MCEOx) enables the corresponding output event. The output event is disabled by writing EVCTRL.MCEOx=0.

One of the following event actions can be selected by the Event Action bit group in the Event Control register (EVCTRL.EVACT):

- Disable event action (OFF)
- Start TC (START)
- Re-trigger TC (RETRIGGER)
- Count on event (COUNT)
- Capture time stamp (STAMP)
- Capture Period (PPW and PWP)
- Capture Pulse Width (PW)

Writing a '1' to the TC Event Input bit in the Event Control register (EVCTRL.TCEI) enables input events to the TC. Writing a '0' to this bit disables input events to the TC. The TC requires only asynchronous event inputs. For further details on how configuring the asynchronous events, refer to *EVSYS - Event System*.

Related Links

EVSYS – Event System on page 544

35.6.7. Sleep Mode Operation

The TC can be configured to operate in any sleep mode. To be able to run in standby, the RUNSTDBY bit in the Control A register (CTRLA.RUNSTDBY) must be '1'. This peripheral can wake up the device from any sleep mode using interrupts or perform actions through the Event System.

If the On Demand bit in the Control A register (CTRLA.ONDEMAND) is written to '1', the module stops requesting its peripheral clock when the STOP bit in STATUS register (STATUS.STOP) is set to '1'. When a re-trigger or start condition is detected, the TC requests the clock before the operation starts.

35.6.8. Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

The following bits are synchronized when written:

- Software Reset and Enable bits in Control A register (CTRLA.SWRST and CTRLA.ENABLE)
- Capture Channel Buffer Valid bit in STATUS register (STATUS.CCBUFVx)

The following registers are synchronized when written:

- Control B Clear and Control B Set registers (CTRLBCLR and CTRLBSET)
- Count Value register (COUNT)
- Period Value and Period Buffer Value registers (PER and PERBUF)
- Channel x Compare/Capture Value and Channel x Compare/Capture Buffer Value registers (CCx and CCBUFx)

The following registers are synchronized when read:

- Count Value register (COUNT): synchronization is done on demand through READSYNC command (CTRLBSET.CMD).

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

38.8.5. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x07
Reset: 0x00
Property:

Bit	7	6	5	4	3	2	1	0
							GFMCMP	ENCCMP
Access							R/W	R/W
Reset							0	0

Bit 1 – GFMCMP: GF Multiplication Complete
This flag is cleared by writing a '1' to it.

This flag is set when GHASH value is available on the Galois Hash Registers (GHASHx) in GCM mode.

Writing a '0' to this bit has no effect.

This flag is also automatically cleared in the following cases.

1. Manual encryption/decryption occurs (START in CTRLB register).
2. Reading from the GHASHx register.

Bit 0 – ENCCMP: Encryption Complete
This flag is cleared by writing a '1' to it.

This flag is set when encryption/decryption is complete and valid data is available on the Data Register.

Writing a '0' to this bit has no effect.

This flag is also automatically cleared in the following cases:

1. Manual encryption/decryption occurs (START in CTRLA register). (This feature is needed only if we do not support double buffering of DATA registers).
2. Reading from the data register (DATAx) when LOD = 0.
3. Writing into the data register (DATAx) when LOD = 1.
4. Reading from the Hash Key register (HASHKEYx).

39.8.7.7. Host Status Pipe

Name: STATUS_PIPE
Offset: 0x0E & 0x1E
Reset: 0xxxxxxxx
Property: PAC Write-Protection, Write-Synchronized, Read-Synchronized

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	ERCNT[2:0]			CRC16ER	TOUTER	PIDER	DAPIDER	DTGLER
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	x	x	x	x	x	x

Bits 7:5 – ERCNT[2:0]: Pipe Error Counter
These bits define the number of errors detected on the pipe.

Bit 4 – CRC16ER: CRC16 ERROR
This bit defines the CRC16 Error Status.

This bit is set when a CRC 16 error has been detected during a IN transactions.

Value	Description
0	No CRC 16 Error detected.
1	A CRC 16 error has been detected.

Bit 3 – TOUTER: TIME OUT ERROR
This bit defines the Time Out Error Status.

This bit is set when a Time Out error has been detected during a USB transaction.

Value	Description
0	No Time Out Error detected.
1	A Time Out error has been detected.

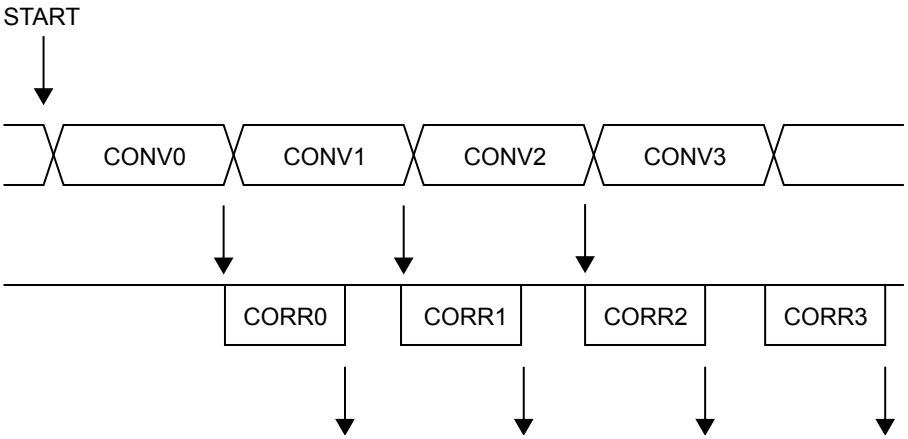
Bit 2 – PIDER: PID ERROR
This bit defines the PID Error Status.

This bit is set when a PID error has been detected during a USB transaction.

Value	Description
0	No PID Error detected.
1	A PID error has been detected.

Bit 1 – DAPIDER: Data PID ERROR
This bit defines the PID Error Status.

Figure 42-8. ADC Timing Correction Enabled



42.6.3. Additional Features

42.6.3.1. Device Temperature Measurement

Principle

The device has an integrated temperature sensor which is part of the Supply Controller (SUPC). The analog signal of that sensor can be converted into a digital value by the ADC. The digital value can be converted into a temperature in °C by following the steps in this section.

Configuration and Conditions

In order to conduct temperature measurements, configure the device according to these steps.

1. Configure the clocks and device frequencies according to the Electrical Characteristics.
2. Configure the Voltage References System of the Supply Controller (SUPC):
   - 2.1. Enable the temperature sensor by writing a '1' to the Temperature Sensor Enable bit in the VREF Control register (SUPC.VREF.TSEN).
   - 2.2. Select the required voltage for the internal voltage reference INTREF by writing to the Voltage Reference Selection bits (SUPC.VREF.SEL). The required value can be found in the Electrical Characteristics.
   - 2.3. Enable routing INTREF to the ADC by writing a '1' to the Voltage Reference Output Enable bit (SUPC.VREF.VREFOE).
3. Configure the ADC:
   - 3.1. Select the internal voltage reference INTREF as ADC reference voltage by writing to the Reference Control register (ADC.REFCTRL.REFSEL).
   - 3.2. Select the temperature sensor vs. internal GND as input by writing TEMP and GND to the positive and negative MUX Input Selection bit fields (ADC.INPUTCTRL.MUXNEG and .MUXPOS, respectively).
   - 3.3. Configure the remaining ADC parameters according to the Electrical Characteristics.
   - 3.4. Enable the ADC and acquire a value, ADC_m.

Calculation Parameter Values

The temperature sensor behavior is linear, but it is sensitive to several parameters such as the internal voltage reference - which itself depends on the temperature. To take this into account, each device contains a Temperature Log row with individual calibration data measured and written during the

43.8.7. Status A

Name: STATUSA
Offset: 0x07
Reset: 0x00
Property: Read-Only

Bit	7	6	5	4	3	2	1	0
			WSTATE0[1:0]				STATE1	STATE0
Access			R	R			R	R
Reset			0	0			0	0

Bits 5:4 – WSTATE0[1:0]: Window 0 Current State
These bits show the current state of the signal if the window 0 mode is enabled.

Value	Name	Description
0x0	ABOVE	Signal is above window
0x1	INSIDE	Signal is inside window
0x2	BELOW	Signal is below window
0x3		Reserved

Bits 1,0 – STATEx: Comparator x Current State
This bit shows the current state of the output signal from COMPx. STATEx is valid only when STATUSB.READYx is one.

44.8.13. Data Buffer DAC0

Name: DATABUF0
Offset: 0x14
Reset: 0x0000
Property: Write-Synchronized

Bit	15	14	13	12	11	10	9	8
	DATABUF[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DATABUF[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – DATABUF[15:0]: DAC0 Data Buffer
DATABUF0 contains the value to be transferred into DATA0 when a START0 event occurs.

45.4. Signal Description

Table 45-1. Signal Description for PTC

Name	Type	Description
Y[m:0]	Analog	Y-line (Input/Output)
X[n:0]	Digital	X-line (Output)

Note: The number of X and Y lines are device dependent. Refer to *Configuration Summary* for details.

Refer to *I/O Multiplexing and Considerations* for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

Related Links

Configuration Summary on page 16
I/O Multiplexing and Considerations on page 30

45.5. Product Dependencies

In order to use this Peripheral, configure the other components of the system as described in the following sections.

45.5.1. I/O Lines

The I/O lines used for analog X-lines and Y-lines must be connected to external capacitive touch sensor electrodes. External components are not required for normal operation. However, to improve the EMC performance, a series resistor of 1kΩ or more can be used on X-lines and Y-lines.

45.5.1.1. Mutual-capacitance Sensor Arrangement

A mutual-capacitance sensor is formed between two I/O lines - an X electrode for transmitting and Y electrode for receiving. The mutual capacitance between the X and Y electrode is measured by the Peripheral Touch Controller.

Figure 45-3. Mutual Capacitance Sensor Arrangement